1 Installation

The installation of the package should be fairly simple. Insert the disk into the diskdrive and enter the diskdrive. If it concerns the A-drive, just type `A:` (for the B-drive `B:` and so on). now enter the command `install` followed by the enter or return key. You are now asked whether the program is to be installed on the C: drive in the subdirectory called RIPE. If this is not what you want, interrupt the installation. You can do this by holding the control (Ctrl) key down and then hitting the 'c'. If on the other hand you have no objection to the fact that RIPE will be installed in the directory `C:\RIPE` just strike the enter key. For installation on other disks or in other subdirectories type `install path` instead of `install` only. For example, `install D:\MIRROR\EPIR` will install RIPE on the D-drive in the subdirectory called `\MIRROR\EPIR`.

2 Your first steps in RIPE: the basics

After you have installed the package you can begin a session. The first thing you have to do is to enter the subdirectory containing the executable file. If you followed the instructions in the **installation** section, this file (called **ripe.exe**) should be in the subdirectory `C:\ripe` of your harddisk. Therefore, being at the prompt `C:\>` you type `cd ripe`.

To start the program type `ripe`. The output screen should now display the name of the program and the time it was compiled. This should be helpful in case new releases of the program come out. You should now follow the programs instructions and type the return or enter key.

If all went well you are now in the main menu. Remember that you can always come back to the main menu from any point in the program by typing enough **0**'s. If you type a zero in the main menu (followed by the return key) the program asks if you really want to quit RIPE. If you have entered a lot of variables, this check ensures that you at least think once again whether or not you want to remove all your results and series from memory. If you quit RIPE without having saved your results, these results cannot be retreived. So be careful! Now try quitting RIPE by typing a zero. You are now prompted to type either an 'n' or a 'y'. As we do not want to quit right now, type an 'n'.

The first thing we need for performing some calculations is data. Therefore type a '1' in order to enter the data manipulation menu. Notice that you can go to the previous menu now by choosing option '0'. This works for most menus in the program. Let us add some variables to the workspace by choosing option 2. There are 5 ways to achieve this objective. We begin with the most simple one: loading a variable from keyboard input. Type '3' followed by return. The program now asks for the number of elements in the series that will be supplied. If you enter a '0' here, the program goes back to the menu without reading the series. (Try it!) However, we will enter a short series containing 5 elements. Type '5'. The program now wants to know the actual numbers. Supply the series 1.1 1.2 1.0 1.3 2.0.

After you have supplied the numbers, the program asks for a name of the variable/series. Let us call it 'test'. RIPE can deal with yearly data, seasonal data, and non time series data. In order to obtain the necessary information you are asked to supply the starting year of the data. If the data has no time component, you can type a 0. It is important however that the number you

supply here be the same for all series that you use in your regression. The reason for this is that RIPE interprets each series as a time series, even if it has no time component. Assume that you enter two series, called test1 and test2. The first begins at time 0, while the other begins at time 1. Both series contain 5 observations. Now if you want to perform a regression of test2 on test1, RIPE will obviously use the last four observations of test1 and the first four of test2. So it performs a regression using only four observations. This automatic sample selection can prove very helpful when working with time series data. When using cross section data however, one has to be aware of its possible pitfalls. You now supply the number 0, but remember that any number will do (only use **the same** number for every cross section series).

Now a similar problem as the one described in the previous paragraph arises. Only in this case it concerns the seasonality in the data. As we have no seasonality, you can supply either a '0' or a '1'. Both entries have the same effect. After having answered these few questions, you are back in the menu for adding variables to the workspace. Notice the difference with the first time you entered this menu. At the top of the screen is now a heading 'Variables' which shows the number of variables in the workspace along with their names.

Suppose we want to estimate the location parameter of test. In order to do that we need a constant term. Choose option 5: generate dummies, constants and linear trends. Now choose option 1, generating an additive sequence. The length of the series is 5, the series starts in year 0 and contains no (=0 or 1) seasons. Call this variable const. Now the actual making of the variable begins. The first value of the series must be 1. Furthermore, because we want to generate a constant, the increment has to be 0. Notice that if you want to generate a trend, than you can supply an increment of 1. Again we are back in the menu for adding variables. There are now two variables in the workspace (=memory). Note that the same pitfalls are present for supplying the first year of a constant or trend as described before. When using non time series data, you must supply the same starting year for all of the series, including constants and trends.

There is also a shortcut to generate constants, which only works well if you are sure there are no zero entries in a series. Choose option 4: creating a new variable be transforming an old one. You can choose between no less than 25 transformation options. Choose 24: var1/var2, and supply the same variable two times. In our case we can use variable number 1. The new variable is

a constant, because $x/x = 1$ for all $x \neq 0$. Call the new variable 'const2'. If you have a constant term in the workspace, you can easily generate a trend by remembering that a trend is nothing more than the cumulative sum of a constant term. Therefore you can use transformation option 7. The advantage of using the technique described above is the following. By dividing a series by itself, the resulting series will have the same number of seasons, starting year and starting season as the old series. So you do not have to answer several additional questions as was the case when using the technique of the preceding paragraph.

Because we now have two constant terms, we have one variable to many in our computer memory. In order to delete it, go to the previous menu by typing a zero. When you are in the menu for adding variables, you again choose zero to reach the menu for data maintenance. There you choose option 3: deleting variables. You can now follow the instructions on the screen. The program always prompts you to answer whether you are sure you want to delete the chosen variable(s). You can confirm your choice by typing 'y' or '1'. Try deleting the third variable. Type 3 followed by 1. As you can see there are now only two variables left in the workspace. Type a 0 to return to the previous menu.

By now you have learned three ways of entering data. You can use keyboard input, generate dummies and trends and transform existing variables. You have also learned how to delete variables from memory. So now it becomes time to learn how to save variables from memory. Choose option 1: print or export workspace variables. First you can inspect the values in the series on the screen. Choose option 1. The program now wants to know which variables you want to display. As you want to display all of them, you supply a negative number (as indicated on the screen). Just type -1. You can now inspect the entries and type a return after each page of numbers. Because our two series are very short, they fit on only one page.

Visual inspection is not the most effective way to store numbers over a longer period. RIPE offers you the choice between two kind of files to store numbers. The first type is the ASCII type. When you choose this option (option 2) the output is the same as that which you saw on the screen earlier. The only difference is that now the output is sent to a file instead of to the screen. Therefore, a filename must be given when you choose this option (along with an extension). If the file already exists, RIPE asks you whether you want to overwrite this existing file, append the present output to it or

quit the printing of the variables altogether. The second type of file you can use is the RIPE-file. RIPE-files have the extension `.alc`, no matter what extension you supply when you enter the filename. The advantage of `.alc` files is that the number of seasons, the begin and end year and the starting season are all saved along with the variables. So when reading such a file, you do not have to enter these numbers each time you load the variables into memory. A disadvantage of these files is that they are not human readable, as opposed to the ASCII files. However, these latter files cannot be directly used for loading variables during a second RIPE-session. This is a consequence of the fact that for example the names of the variables are also printed in this file.

Now let us export our two variables to the RIPE-file `test.alc`. Choose option 3 and enter -1 to select all variables (you can also enter a 1, 2 and 0 consecutively). The variables are now saved to disk. To look at the result, enter 0 to go to the previous menu. Enter 0 again to arrive at the main menu. Now choose option 4: directory manipulation. If you choose option 2, you can see that the file `test.alc` has been made. Follow the instructions and type a 'c' followed by return to go back to the menu. Option 1 of the menu needs no further explanation. Just try it if you want. Option 3 requires you to supply a mask. For example, supplying the mask `*.exe` will display all executable files on the screen, while `a*.*` will display all files starting with an `a`. Now enter 0 again to go to the main menu.

In order to illustrate the advantages of the use of `.alc` files, delete all variables from memory. Enter the data manipulation menu by typing 1, choose option 3, supply a -1 to delete all variables and confirm your choice with a 'y'. Notice that RIPE now automatically leaves the menu for deleting variables. If you try to reenter it, you will be warned that this is not possible, the logic being that you cannot delete that which is not there. Now try adding variables from `.alc` files by choosing the option 2 twice. As you remember the name of the file is `test.alc`. However, because RIPE-files always carry the extension `.alc`, you can only type `test`. Note however that typing `test.alc` or `test.xxx` has the same effect. The extension is just ignored. Once you have supplied the filename, the variables contained in the file are displayed, along with the time period during which they are defined. You can load variables into memory in the ordinary way. Just type the number in front of the variables you want to load. If you want to load all of them, just supply a -1. If you want to load none, enter a 0. You also have to supply a 0 if you

do not want to load any other variables into memory after you have already selected several of them. Now go back to the main menu.

By now it is high time to introduce you to the regression part of the program. Choose option 2 from the main menu. RIPE always works with the model $y = X\beta + u$, where y is the variable to be explained and X is the matrix containing the regressors. In order to perform a regression, the y and X matrices must be constructed. Choose option 1. The program asks you to supply the regressors. If you are finished with supplying them, you just enter a zero. In our case only the constant is a regressor, so you first choose 2 and then 0. RIPE then prompts for the independent variable (look on your screen!). In our case this is variable number 1. The program now returns to the model construction and estimation menu. Note the difference with the previous time you entered this menu. At the top are now the names of the independent variables and of the dependent one.

You can calculate the mean of test by performing an OLS regression of `test` on `const`. Choose option 4. The output should look something like this:

```
*************************************************************
Output for UNWEIGTHED OLS

The independent variable is test
---------------------------------------
               const        1.3200    (   0.1772)        7.4492

The data contains no seasonality and starts in 0
number of OBSERVATIONS = 5
Scale estimate =     0.3962
Variance estimate =     0.1570


*************************************************************
```

The number 1.32 is the estimated mean of test. The standard error of this estimate equals 0.1772, which results in a t-value of 7.4492. There is no seasonality in the series, the number of observations is 5 and the standard error and variance of the regression are also supplied. After each regression

you are prompted to answer three questions. Because these questions do not have a dual interpretation I leave it to you to experiment for yourself.

By now you have learned the basics of RIPE. You should be able to create, load and transform variables. Also deleting variables, showing them on the screen and saving them to disk should pose no problems. In short, you should now be able to perform ordinary least squares regressions with RIPE. You can either stop reading now and experiment further with the program for yourself, or you can continue reading the next section, entitled 'Carrying on: robust estimation'.

3 Carrying on: robust estimation

In the previous section you were introduced to the techniques for performing ordinary least squares regressions with RIPE. As is well known in the literature, the ordinary least squares estimator (OLS-estimator from now on) is very sensitive to the occurrence of outliers. I will not attempt to introduce you to the vast literature on robust statistics. If you are interested you can start with the few references given at the end of this manual. In order to overcome the vulnerability of the OLS-estimator to outliers, one can either use a lot of diagnostic measures or take refuge to robust estimation techniques. Many diagnostic measures are nowadays incorporated into standard statistical programs like SAS. If you use one of these programs it would be a waiste if you did not at least take a look at the diagnostics associated with your estimated equation. On the other hand very few of the main statistical packages incorporate sophisticated robust estimation techniques. Several alternative programs are available, see for example section 6.4 of Hampel et al. (1986). RIPE just amends another program name to this list.

In order to perform some serious experiments two well known data sets are supplied along with the package. The first set of data gives the number of international telephone calls from Belgium. The second set contains data of the star cluster CYG OB1. The data along with a more elaborate description can be found in e.g. Rousseeuw and Leroy (1987, p. 25-28).

We are now going to perform a robust regression on the first data set. First remove all resident series from memory. Then enter the menu to enter variables and select the option to load data from a RIPE-file. This is option 2 from the 'add a variable to the list' menu. Now supply the name **telephon**. The program now displays something like this:

```
Give the name of the file ? telephon
There are 2 variables in this file
 1: time from 1 to 24
 2: tel.calls from 1 to 24
Enter a negative number if you want to load all these variables.
Enter zero to quit.
Include which variable ?
```

You see that both the names of the variables and their range are stored in the RIPE-file. Because you need both variables enter a negative number,

e.g. -1. You also need a constant term in the regression later on. Therefore use one of the techniques of the previous section to generate one. Give this constant term the name `const`. After you have done all of this, go back to the main menu (by entering enough zeros).

Once back in the main menu, choose option 3 in order to inspect the data visually. Our objective is to get a plot on the screen of the number of telephone calls against time, with time being plotted on the horizontal axis. So when the program asks for the series on the x-axis, you can supply a 1 in order to use the real time. On the other hand you can also supply a 0. In that case the y-series will be plotted against their index number. This has almost the same effect, only the numbers given on the x-axis will differ.

The program now asks for the series to be plotted against the given x-series. If you supply a zero directly, you will leave the plotting routine without plotting anything. You could also have achieved this by supplying a negative number when the program prompted you to supply an x-series. If you want to see a real plot however, choose the number of the series you want to plot. In our case this is the number 2 of the telephone calls series. After you have supplied this number the program asks you for a second series (and a third, and so on). However, we only want to plot one series, therefore you supply a 0 in order to stop entering y-series.

Next you have to answer several questions. The first is whether you want a line plot (using different numbered or colored lines for different series), a scatter plot (using different symbols for different series), or a scatterplot using index numbers for each point of the series instead of identical symbols. Choose option 3 (or experiment with the other options yourself). You now have to supply a title, x-label and y-label for the plot. The least you have to supply when answering these questions is a space-character followed by the enter key. In order to show you the final result however, I supplied `title`, `x-label` and `y-label`. The final plot will look something like figure 1. If you use the graphical version of ripe, it will of course look a bit different. This is how you should read the axes. A $-$ in front of the number indicates that the number is negative. At the end of each number you again see a sign ($+$ or $-$) followed by a single digit. The number to the left of these two characters has to be multiplied by 10 to the power (sign and digit). So $-.150+0$ means $-0.150 * 10^{+0} = -0.150$, while $.617 - 3$ means $0.617 * 10^{-3} = 0.000617$.

You can easily see that the observations 15 to 20 (and perhaps 21) are in disagreement with the pattern given by the remaining data points. As

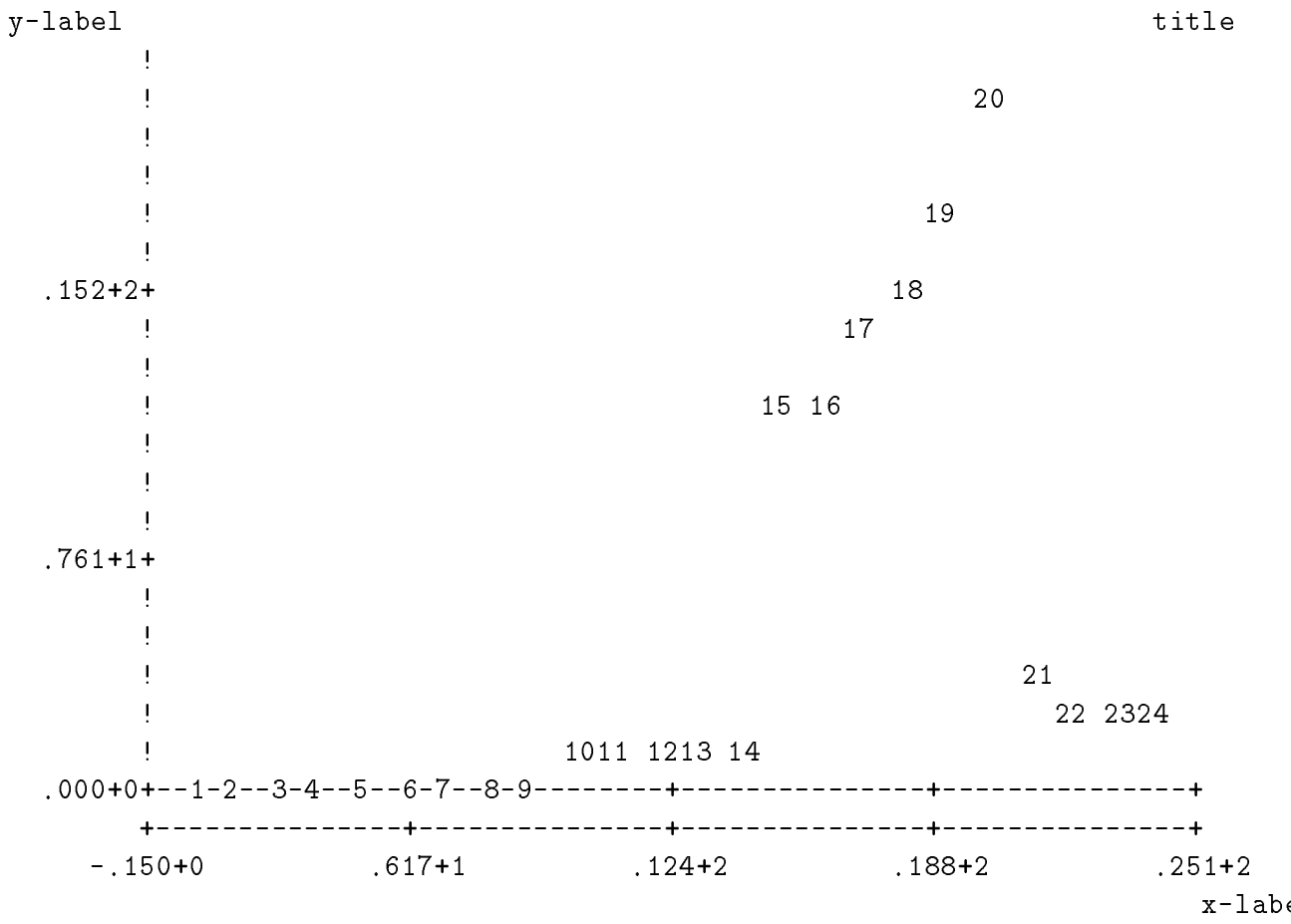

Figure 1: Making a plot in RIPE

is explained by Rousseeuw and Leroy, these observations are outliers in the sense that they are gross measurement errors. Therefore it seems a sensible strategy to omit them from the regression. However, we will let them stay in the sample in order to demonstrate the effectiveness of several robust methods. This corresponds to a situation in which one does not know exactly which points are the outliers. You can think of performing a regression on several explanatory variables. In such situations it becomes difficult to identify the outliers in such a simple way as above (if they can be identified at all). In order to end the plot type the return key. You are then back in the main menu.

Choose option 2 of the main menu to build a regression model. You start by constructing the X and y matrix. Let the X matrix consist of a trend and a constant, and let the number of telephone calls be the y variable. If you perform an OLS regression you get a value of -1.3027 (2.3690) for the constant and of 0.5041 (0.1658) for the trend coefficient. These values are heavily influenced by the outliers. In order to show this, we calculate a second regression estimate. Choose option 5 from the model construction menu. You are now about to perform the a 'very' robust LMS regression. RIPE displays first the number of elemental subsets for the present regressors, which is 276. It then asks whether you want to use all these sets. As the number is not very large, you can answer yes. If the number becomes very big however, as is the case with many regressors and many cases, you should answer no and supply the number of (randomly chosen) elemental subsets you actually want to use. The computer now starts number crunching. You are being notified how many elemental subsets have been examined so far. The final result is:

```
Constant should be corrected with   -0.00231
************************************************************
Regression output for LMS

The independent variable is tel.calls
----------------------------------------
                time        0.1154
               const        0.0462

The data contains no seasonality and starts in 1
number of OBSERVATIONS = 24
```

```
Scale estimate =      0.1164
Variance estimate =     0.0135

************************************************************
```

Just note the difference with the OLS estimates. In order to inspect which cases are outliers, you can store the residuals and devide them by the scale estimate 0.1164. Print these scaled residuals and look which of them are greater than for example 2.5. At the top of the regression output is a number indicating by how much the constant, if present, should be adjusted for a better fit yet. The fact that the actual constant is not adjusted is a result from the fact that RIPE cannot for itself decide which column of the X matrix is the constant. A second thing to note is the absence of standard errors and t-values. This is a result of the poor asymptotic properties of LMS. However, the LMS estimator is very usefull as a diagnostic tool and as a starting point for more refined robust estimators, like the S estimator of Rousseeuw and Yohai (1984) and the MM-estimator of Yohai (1987). Each of these procedures must start from a very robust estimate like the LMS-estimate.

The MM-estimator needs no further explanation. Just experiment with it for yourself. The cutoff number you have to supply (which has to be positive) determines the degree of robustness. The lower this number, the more robust will be the estimate. On the other hand you must remember that lowering this number also increases the asymptotic standard errors of the estimates, which is something you do not want. The default cutoff value of 4.685 gives you approximately 1.05 times larger asymptotic standard errors than the OLS estimator, at least in case you had an outlier free sample.

The S-estimator again requires you to supply a cutoff value (which this time determines the efficiency (standard errors) and the breakdown point (percentage of outliers the estimator can cope with). Moreover, you must also supply a maximum number of iterations within which convergence must be achieved. If you have used a very robust starting estimate, there should be no problem in choosing a fairly large value (say 50). But if you do not use such a preliminary estimate I think it is safer to supply a small number here and use the final estimate as the starting point for a new S-estimate. This procedure can be repeated several times until you are sure that you are in the

right area of the parameter space. You can then supply a larger maximum number of iterations. It is possible that the S-estimator hangs for a while, taking very small steps. It is the experience of the author that almost always the routine will after a while take up bigger step sizes, reaching lower values of the objective function (even in case the objective function rises for a while when small steps are taken). So far about high breakdown estimation. The next section considers M-estimators.

4 Proceeding further: more about robust estimation

In this section we work with a different data set. If their are any variables in memory, remove them first. Then enter the menu to load variables and extract all the variables that are in the RIPE-file called **stars**. The data set contains a constant term and 47 pairwise observations of stars, namely the log light intensity and the log surface temperature of the stars. If you make a scatter plot of log light intensity versus log temperature, you can easily see that there are several drastic outliers. The four stars in the corner of the plot are red giants, which are extremely bright with respect to their temperature.

Next go to the model construction menu and make an X matrix containing a constant term and the log temperature. The y matrix on the other hand must contain the log light intensity of the stars. Using this X and y for performing an OLS regression gives you the following coefficients. The constant equals 6.7935, while the coefficient for the log temperature equals -0.4133. Now store the fitted values of this regression and make a scatter plot with the log temperature on the horizontal axis and with the log lightintensity and fitted OLS series on the vertical axis. Your plot should look like figure 2. The 1 indicates the original data, while the 2 indicates the OLS fit. You can see that the line found by OLS is not at all like the linear pattern indicated by the bulk of the data. This last pattern indicates that the line should have a positive instead of a negative slope coefficient. The apparent discrepancy is due to the several outlying giant stars. One possible remedy you could think of is to use an M-estimator instead of the OLS-estimator. RIPE provides facilities that allow you to do this.

In order to use an M-estimator you must first choose one out of several ψ-functions. Choose option 3 from the model construction menu. You can now choose between 6 functional specifications. The most commonly used ones are the Huber and Bisquare ψ-function. Try choosing the bisquare function, which is redescending. The program then asks you to supply a cutoff value. For the bisquare function the value 4.685 is often used, so supply this value. After you have done this, you can choose option 8, which makes the computer perform the regression with the chosen M-estimator. The program uses the algorithm of iterative weighted least squares to do the actual computations. The result is a constant of 6.8367 and a log temperature coefficient of -

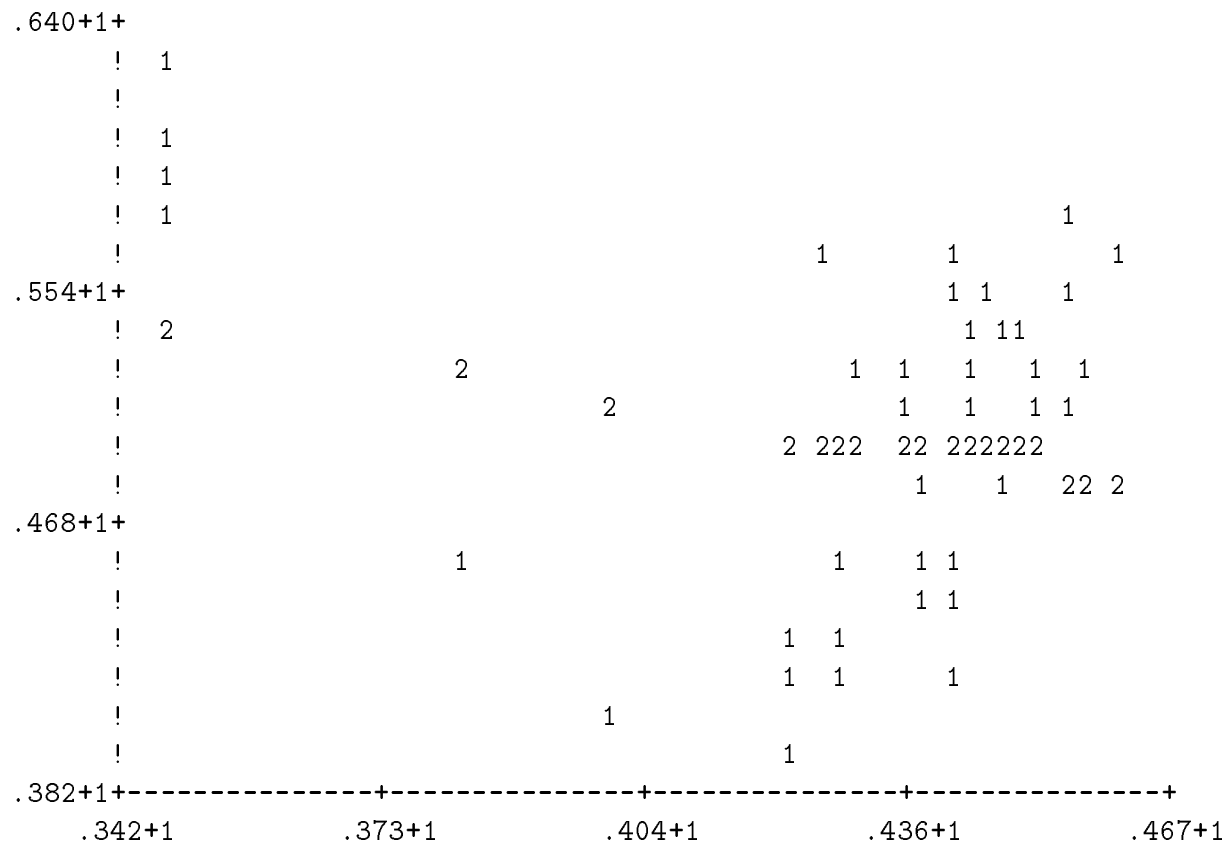

Figure 2: The stars data and OLS fit

0.4203. Note that this latter quantity is still negative. This phenomenen is well known in the literature. M-estimators are known to be able to cope with outliers in the vertical direction (in the y), but also to be vulnerable to outliers in the space of explanatory variables (in the X). As you can see in the scatter diagram you have made, the latter situation is of importance in our case.

It is precisely for the case of outlying values of the X variables that generalized M-estimators (or GM-estimators) were proposed. The main difference with M-estimators is that a weighting matrix must be specified for the X matrix. In real life situations you are only uncertain about outliers in the non-deterministic columns in X. For example, if there is a constant or a trend (or a dummy) in the X matrix, you know that these regressors do not cause outlying values of X. So when constructing weights for X, you must first remove these columns. Choose option 2 from the model construction menu. Then choose option 1: deleting a variable from the X matrix. Next choose the constant term. Your X matrix now only contains the log temperature. By entering a 0 you go back to the main model construction menu.

You now have to compute weights for the present X matrix. Start by choosing option 10 from the model construction menu. You can choose between several distance measures of rows of X from some center. As the first two choices, OLS leverages and Mahalanobis distances, are nonrobust and mainly meant for diagnostic purposes, I will focus on the two MVE (=Minimum Volume Ellipsoid) options. Details for the computation can be found in the article of Rousseeuw and van Zomeren (1990). The MVE-estimator is a kind of multivariate generalization of the LMS-estimator discussed earlier. The total enumeration grows out of hand quite rapidly, demanding very long computation times. The random sampling approach can be used as an alternative to approach the MVE-estimator. A second approximation is given by using the second algorithm of the article mentioned above. Both approximations have their limitations.

Choose option 3, the random sampling approach. You can use for example 100 subsamples. After the number crunching, several results are shown. Let them roll over your screen by following the hints supplied by the program. After these results you are asked whether you want to store the computed distances, either as raw distances or as weights. In order to perform a GM-regression, you need these weights, so answer yes. Call the variable `weights` or something like that. The program then shows how weights are (optionally)

constructed from `weights`. You are asked whether you want to store the untransformed distances (option 'raw') or the transformed (into weights) ones (option 'weights'). Store them as weights. Similar questions must be answered for each of the other options from this menu. Now go up one menu to the model construction and estimation menu.

Once arrived at this menu, you now choose option 11: choosing weights. The list of variables is presented and you can select the constructed `weights` series by selecting its number. You are immediately back at the model menu. Note the difference, because at the top of the screen there is a message saying that weights are specified. With these weights you can perform weighted least squares regression. But before you do that, the constant term must be added again to the matrix containing the explanatory variables. Choose option 2, and then option 2 again to add a variable. Choose the constant term. The program asks if you want to remove the weights. Of course you do *not* want this. The question is however useful in case you add a variable which can contain outliers, because then it would be wise to calculate the weights for the new X matrix anew. However, in our simple example everything is ok (concerning the constant).

Go back to the model construction menu and perform a weighted least squares regression by choosing option 4. The constant term now equals 0.2852 and the coefficient of log temperature is 1.0622. This already turns out to be a major improvement. Perhaps we can improve even more by computing a GM-estimate. Choose option 8 and note the huge difference still with the weighted least squares estimates. The constant equals -7.1259 and the other coefficient is 2.7396. If you want to compute a one-step (or fixed-k-step) (G)M-estimator, choose option 9 and first enter the number of steps.

5 Concluding remarks

By now you have learned most there is to know about RIPE. I hope you will enjoy using the program. Remember, all comments are welcome at the address mentioned in the preface.